

7. Aritmetični izrazi

1

Aritmetični izrazi

2

- aritmetične izraze tvorimo s pomočjo aritmetičnih operatorjev
- aritmetični operatorji
 - + - * / %
(seštevanje, odštevanje, množenje, deljenje, modulo)
- modulo – celoštevilski ostanek pri deljenju
 - v matematiki: $a \bmod b$
 - npr. $7 \% 2 = 1$, ker je $7 = 3 * 2 + 1$
- z operatorji lahko ločimo identifikatorje
 - med njimi uporabimo lahko tudi nevidne znake, ni pa nujno

Prioriteta in asociativnost operatorjev

3

- operatorji se obnašajo v skladu s pravili prioritete in asociativnosti
 - prioriteta – prednost glede na druge operatorje
- vrstni red vrednotenja lahko spremenimo z oklepaji
 - z oklepaji tudi olajšamo čitljivost izrazov

PRIMERI:

$1+2*3$ se ovrednoti na 7, saj ima * prednost pred +
 $1+(2*3)$ kot prej
 $(1+2)*3$ se ovrednoti na 9
 $1+2-3+4-5$ ker imata + in - isto prioriteto, se vrednosti z leve proti desni

Tabela prioritete operatorjev in asociativnost (1)

4

prioriteta	operatorji	asociativnost
najvišja	:: (globalni ali razredni doseg)	od leve k desni (LD)
	func() [] -> . (pripona)++ (pripona)-- typeid(e) type(e) dynamic_cast<tip>(e) static_cast<tip>(e) reinterpret_cast<tip>(e) const_cast<tip>(e)	LD
	++(predpona) --(predpona) ! ~ &(naslov) sizeof(e) +(unarni) -(unarni) *(indirection) delete new	od desne k levi (DL)
	.* ->*	LD
	* / %	LD
	+ -	LD
	<< >>	LD

Tabela prioritete operatorjev in asociativnost (2)

5

prioriteta	operatorji	asociativnost
	<= < > >=	LD
	== !=	LD
	&	LD
	^	LD
		LD
	&&	LD
		LD
	?:	DL
	= += -= *= /= %= >>= <<= &= ^= =	DL
	throw(e)	LD
najnižja	,	LD

Še o operatorjih

6

- opomba: unarni operatorji imajo prednost pred binarnimi
 - `-a*b-c` `(-a)*b-c`
- rezultat aritmetičnih operacij je odvisen od argumentov
 - `a=3/2` `// se ovrednosti na 1`
 - `a=3/2.0` `// se ovrednosti na 1.5`

Mešani izrazi

7

- izrazi, v katerih so argumenti različnih tipov
 - rezultat je enakega tipa kot "največji" med tipi argumentov
- splošna pravila za samodejna pretvarjanja
 1. Vsak **bool**, **char**, **short** ali **enum** se poviša v **int**.
Celoštevilске vrednosti, ki jih ne gre predstaviti kot **int**, se pretvorijo v **unsigned**.
 2. Če je po prvem koraku izraz še vedno mešani, velja naslednja hierarhija tipov:

$$\text{int} < \text{unsigned} < \text{long} < \text{unsigned long} < \text{float} < \text{double} < \text{long double}$$
- operand nižjega tipa se poviša v višji tip
 - vrednost izraza je potem tega tipa

Mešani izrazi - primeri

8

- v primeru naj veljajo naslednje deklaracije:
`char c; long lg; double d; short s; float f; unsigned u; int i;`

izraz	tip	izraz	tip
<code>c-s/i</code>	<code>int</code>	<code>u*3-i</code>	<code>unsigned</code>
<code>u*3.0-i</code>	<code>double</code>	<code>f*3-i</code>	<code>float</code>
<code>c+1</code>	<code>int</code>	<code>3*s*lg</code>	<code>long</code>
<code>c+1.0</code>	<code>double</code>	<code>d+s</code>	<code>double</code>

Pretvarjanje tipov

9

- povišanje oz. razširitev tipa ne predstavlja težav
 - `d = i;`
- degradiranje oz. zožitev tipa lahko izgubi informacijo
 - `i = d;`
- zraven posrednih (samodejnih) pretvorb pri prirejanju in mešanih izrazih poznamo direktno oz. eksplicitno pretvarjanje

```
static_cast<double>(izraz);
```

PRIMER:

```
int i = 10;  
double x = static_cast<double>(i)/2;  
double y = double(i)/2;
```